WHAT IS CLAIMED IS:

1.   A source code line counting system, comprising:

a computer readable storage medium; and

a software counting tool stored on the computer readable storage medium and

5   operable to

parse a first file containing computer source code to create a token

stream in response to one of a plurality of sets of configuration data,

wherein the computer source code in the file was written in one of a

plurality of computer languages that may be processed by the software counting tool,

10   wherein each set of configuration data comprises keywords for one or

more of the plurality of computer languages,

create a list of statements in response to the token stream, and

generate a count value in response to the list of statements.

15   2.   The system of Claim 1, wherein the plurality of sets of configuration

data includes a different set for each of the following computer programming

languages: C++ and Cobol.

3.   The system of Claim 1, wherein the plurality of sets of configuration

20   data includes a different set for at least two of the following computer programming

languages: C, C++, Java, Fortran, Cobol, and Basic.

4.   The system of Claim 1, wherein each token in the token stream

comprises a string of one or more characters and wherein each token in the token

25   stream is associated with a token type value.

5.   The system of Claim 4, wherein the token type value comprises a data

value indicating that the token is of a type selected from the group comprising: an

operator, a comment, a constant, a keyword, and an identifier.

30

6.   The system of Claim 4, wherein the possible token type values that can

be chosen do not vary based upon the language that the computer source code in the

first file was written in.

7.   The system of Claim 1, wherein portions of the token stream are ignored in generating the statement list because such portions are associated with a portion of the source code in the first file that is written in a language different from the one of the plurality of computer languages.

8.   The system of Claim 1, wherein each statement in the list of statements is further associated with a statement type value.

9.   The system of Claim 8, wherein the possible statement type values that can be chosen do not vary based upon the language that the computer source code in the first file was written in.

10.   The system of Claim 8, wherein the statement type value comprises a data value indicating that the statement is of a type selected from the group comprising: data, executable, and compiler.

11.   The system of Claim 1, wherein the statement list is created by examining the relationship of a token in the token stream to other tokens preceding or following the token in question.

12.   The system of Claim 1, wherein the software counting tool is further operable to:

parse a second file containing computer source code to create a second token stream in response to the one of the plurality of sets of configuration data,

wherein the second file comprises a different version of the source code contained in the first file,

create a second list of statements in response to the second token stream, and

compare the first list of statements to the second list of statements to generate at least one count responsive to differences between the first list of statements and the second list of statements.

13.   A method of counting lines of source code, comprising:

selecting one of a plurality of sets of configuration data,

        wherein each set of the plurality of sets of configuration data is associated with at least one computer language,

5   wherein, collectively, the plurality of sets of configuration data are associated with a plurality of computer languages,

        wherein the selected set of configuration data comprises the keywords for a first computer language,

        parsing a first file containing computer source code written in the first

10   computer language to create a first token stream in response to the selected set of configuration data,

        creating a first list of statements in response to the first token stream, and

        generating a count value in response to the first list of statements.

15   14.   The method of Claim 13, wherein the plurality of sets of configuration data includes a different set for each of the following computer programming languages: C++ and Cobol.

        15.   The method of Claim 13, wherein each token in the token stream

20   comprises a string of one or more characters and wherein each token in the token stream is associated with a token type value.

        16.   The method of Claim 15, wherein the token type value comprises a data value indicating that the token is of a type selected from the group comprising:

25   an operator, a comment, a constant, a keyword, and an identifier.

        17.   The method of Claim 15, wherein the possible token type values that can be chosen do not vary based upon the language that the computer source code in the first file was written in.

30

        18.   The method of Claim 15, wherein the plurality of sets of configuration data includes a different set for each of at least five different programming languages.

19.   The method of Claim 13, wherein each statement in the list of statements is further associated with a statement type value.

20.   The method of Claim 19, wherein the possible statement type values that can be chosen do not vary based upon the language that the computer source code in the first file was written in.

21.   The method of Claim 17, wherein each statement in the list of statements is further associated with a statement type value and wherein the possible statement type values that can be chosen do not vary based upon the language that the computer source code in the first file was written in.

22.   The method of Claim 13, further comprising:

parsing a second file containing computer source code to create a second token stream in response to the one of the plurality of sets of configuration data,

wherein the second file comprises a different version of the source code contained in the first file,

creating a second list of statements in response to the second token stream, and

comparing the first list of statements to the second list of statements to generate at least one count responsive to differences between the first list of statements and the second list of statements.

23.   A source code line counting system, comprising:

a computer readable storage medium; and

a software counting tool stored on the computer readable storage medium comprising:

5   a plurality of configuration files, each associated with one or more of a plurality of computer languages,

a tokenizer operable to parse a file containing computer source code written in a first computer language to create a token stream,

wherein the parser is operable to parse source code written in

10   any of the plurality of computer languages,

wherein the parser creates the token stream in response to the configuration file associated with the first computer language;

a first statement builder operable to create a list of statements in response to the token stream, and

15   a counter operable to generate a count value in response to the list of statements.

24.   The system of Claim 23, further comprising:

a plurality of additional statement builders wherein each of the plurality of

20   statement builders are associated with one or more computer languages and operable to generate a statement list in response to a token stream generated from a source code file written in the associated one or more computer languages.

25.   The system of Claim 23, wherein each token in the token stream

25   comprises a string of one or more characters and wherein each token in the token stream is associated with a token type value.

26.   The system of Claim 25, wherein the possible token type values that can be chosen do not vary based upon the language that the computer source code in

30   the first file was written in.

27.   The system of Claim 25, wherein each statement in the list of statements is further associated with a statement type value.

28.   The system of Claim 27, wherein the possible statement type values that can be chosen do not vary based upon the language that the computer source code in the first file was written in.

5

29.   A method of measuring changes in source code comprising:

parsing a first file containing computer source code to create a first token stream,

creating a first list of statements in response to the first token stream,

5   parsing a second file containing computer source code to create a second token stream,

creating a second list of statements in response to the second token stream, and

comparing the first list of statements to the second list of statements to generate at least one count responsive to a comparison between the first list of

10   statements and the second list of statements.

30.   The method of Claim 29, wherein the at least one count is equal or approximately equal to a total number of statements in the second list of statements that are modified versions of statements in the first list of statements.

15

31.   The method of Claim 29, wherein the at least one count is equal or approximately equal to a total number of statements on the second list of statements that did not appear on the first list of statements.

20   32.   The method of Claim 29, wherein the at least one count is equal or approximately equal to a total number of statements on the first list of statements that did not appear on the second list of statements.

33.   The method of Claim 29, wherein the at least one count is equal or
25   approximately equal to a total number of statements on the first list of statements that also appear on the second list of statements.

34.   The method of Claim 29, wherein the second file comprises a modified version of the first file and wherein the comparing step generates data approximating or equal to each of the following:

a total number of statements in the second list of statements that are modified

5   versions of statements in the first list of statements,

a total number of statements on the second list of statements that did not appear on the first list of statements,

a total number of statements on the first list of statements that did not appear on the second list of statements, and

10   a total number of statements on the first list of statements that also appear on the second list of statements.

35.   A source code line counting system, comprising:

a computer readable storage medium; and

a software counting tool stored on the computer readable storage medium and operable to

5   parse a first file containing computer source code to create a first token stream,

create a first list of statements in response to the first token stream,

parse a second file containing computer source code to create a second token stream,

10   create a second list of statements in response to the second token stream, and

compare the first list of statements to the second list of statements to generate at least one count responsive to a comparison between the first list of statements and the second list of statements.

15

36.   The system of Claim 35, wherein the second file comprises a modified version of the first file and wherein the comparing step generates data approximating or equal to each of the following:

a total number of statements in the second list of statements that are modified

20   versions of statements in the first list of statements,

a total number of statements on the second list of statements that did not appear on the first list of statements,

a total number of statements on the first list of statements that did not appear on the second list of statements, and

25   a total number of statements on the first list of statements that also appear on the second list of statements.

37.   A source code line counting system, comprising:

a computer readable storage medium;

a plurality of sets of configuration data, each set associated with at least one computer language, the plurality collectively associated with different computer languages; and

a software counting tool stored on the computer readable storage medium and operable to

receive a source code file, and

compute a statistical measure related to the number of lines of source code in the source code file using the one of the plurality of sets of configuration data associated with the computer language that the source code in the source code file was written in.